

University of Nevada, Las Vegas Computer Science 477/677 Fall 2025

Study for Examination February 11, 2025

1. Fill in the blanks.

(a) Any comparison-based sorting algorithm on a file of size  $n$  executes \_\_\_\_\_ comparisons. (Use  $\Omega$  notation.)

(b) Name two well-known divide-and-conquer sorting algorithms.

\_\_\_\_\_

\_\_\_\_\_

(c) The asymptotic time complexity to find an item in an ordered array of length  $n$  is \_\_\_\_\_, using the divide and conquer algorithm \_\_\_\_\_.

(d) In a priority queue, each item in the structure represents an \_\_\_\_\_.

(e) Three kinds of priority queues we've mentioned in class are \_\_\_\_\_, \_\_\_\_\_, and \_\_\_\_\_.

(f) The height of a binary tree with 10 leaves must be at least \_\_\_\_\_. Exact answer please: no partial credit.

(g) "pop" and "push" are operators of \_\_\_\_\_.

(h) "fetch" and "store" are operators of \_\_\_\_\_.

2. Fill in each blank. Write  $\Theta$  if that is correct; otherwise write  $O$  or  $\Omega$ , whichever is correct. Recall that  $\log$  means  $\log_2$ .

(a)  $\log n^2 = \text{_____}(\log n^3)$

(b)  $\log(n!) = \text{_____}(n \log n)$

(c)  $\sum_{i=0}^{n-1} i^k = \text{_____}(n^k)$

(d)  $2^{\log^2 n} = \text{_____}(n^n)$

(e)  $\log n = \text{_____}(\ln n)$

3. True or False.

(i) \_\_\_\_\_ A good programmer would never use an unordered list as a search structure.

(ii) \_\_\_\_\_ If  $\log F(n) = \Theta(\log G(n))$ , then  $F(n)$  must be  $\Theta(G(n))$ .

4. Find the asymptotic time complexity of each of these code fragments in terms of  $n$ , using  $\Theta$  notation.

(a) `for(float x = n; x > 1; x = x/2)`

(b) `for(int i = 0; i*i < n; i++)`

(c) `for(float x = n; x > 2.0; x = sqrt(x))`

(d) `for(int i = 1; i < n; i++)  
 for(int j = 1; j < i; j=2*j)`

(e) `for(int i = 1; i < n; i++)  
 for(int j = i; j < n; j=2*j)`

5. A stack of integers could be implemented in C++ as a linked list as follows.

```
struct stack  
{  
    int item;  
    stack*link;  
};
```

Finish writing the code for the operators push, pop, and empty, below.

```
void push(stack*&s,int newitem)
```

```
bool empty(stack*s)
```

```
int pop(stack*&s)  
{  
    assert(          // what do you need to assert?
```

6. Let  $W_1, W_2, \dots$  be the Wonderful numbers. Similarly to the Fibonacci numbers, each Wonderful number is twice the sum of the previous two Wonderful numbers. The first few Wonderful numbers are 1, 2, 6, 16, 44,  $\dots$ . Find a constant  $K$  such that  $W_n = \Theta(K^n)$ . Show your work.

7. The following portion of C++ code contains an array implementation of queue. Fill in the missing code for the operators “enqueue” and “empty.”

```
struct queue
{
    int A[N]; // N is a constant large enough to prevent false overflow.
    int rear = 0;
    int front = 0; // initially the queue is empty
};

void enqueue(queue&q,int newitem) // inserts newitem into q

bool empty(queue&q) // returns true if q is empty, false otherwise

int dequeue(queue&q) // returns an item from q and deletes that item
{
    assert(not empty(q));
    int rslt = q.A[q.front];
    q.front++;
    return rslt;
}
```

8. Exactly one of the following is true for a binary tree with  $n$  nodes. Which one?
- The height of the tree is  $O(\log n)$ .
  - The height of the tree is  $\Theta(\log n)$ .
  - The height of the tree is  $\Omega(\log n)$ .
9. The items B, Q, T, M, A, X, L, are inserted into a binary search tree in that order.
- (a) List the items of the tree in preorder.
  - (b) List the items of the tree in inorder.
  - (c) List the items of the tree in postorder.
  - (d) List the items of the tree in level order.

10. What follows is part of a C++ implementation of binary tree of integer. (It runs, I checked.) Fill in the missing code for the recursive functions `insert`, and `find`.

```
struct treenode;
typedef treenode*tree;
struct treenode
{
    int item;
    tree left;
    tree right;
};

tree mainroot; // the root of the binary search tree

void insert(tree&root,int n)
    // inserts n into the binary search tree. Duplicates are not inserted.
{
    if(root == NULL)
    {
        root = new treenode;
        root->item = n;
    }
    // 3 lines missing

}

bool find(tree root, int n)
    // returns true if some node contains n, false otherwise.
{
    // 4 lines missing

}
```

11. A min-heap implemented as a binary tree which is, in turn, implemented as an array, is shown in the first row of the array below. In the subsequent rows, show the evolution of the heap when “B” is inserted. (The first column stores the size of the heap. How does that effect your computaton?) You may need to add additional rows.

9	A	H	C	M	N	F	G	P	R	K

12. Rewrite the infix expression  $a - -b * c * (d + -e) * f$  in prefix notation, and then in postfix notation. Don't forget that in prefix and postfix notation, we use “-” for subtraction, but we use “~” for negation.

13. what well-known algorithm does the procedure `mystery1` implement? What well-known algorithm does `mystery2` implement? Assume that A has been declared as `int A[n]`.

```
void swap(int x, int y)
{
    int z = x;
    x = y;
    y = z;
}

void mystery1()
{
    bool finshd = false;
    while(not finshd)
    {
        finshd = true;
        for(int i = 0; i < n-1; i++)
            if(A[i+1] < A[i])
            {
                finshd = false;
                swap(A[i],A[i+1]);
            }
    }
}
```

```
void mystery2()
{
    for(int i = 0; i < N; i++)
        for(int j = i+1; j < N; j++)
            if(A[j] < A[i]) swap(A[i],A[j]);
}
```

14. Solve these recurrences, giving the asymptotic answer in terms of  $n$ , using  $\Theta$  notation.

(a)  $F(n) = 2F(n/2) + 3F(n/3) + n^2$ ;

(b)  $F(n) = 8F(n/2) + n^3$ ;

(c)  $F(n) = F(3n/5) + F(4n/5) + n$

15. Show a circular queue with dummy node items B, M, Q, R, in that order, from front to rear. then show how the queue changes when you insert H, and then execute dequeue.